

VI PARTE CURSO CODIGO MAQUINA

ANALISIS:
La cinta como soporte de información

REGALO:
Un Key Panel todos los meses

MONSER... cada dia mas.

Key Panel
plantillas troqueladas especiales para colocar sobre el teclado de Spectrum.

Games Board
carcasa moldeada para colocar sobre el teclado del Spectrum, con pivotes móviles para indicar las teclas a utilizar en cada programa.

Revista con cassette de juegos de aparición mensual con programas didácticos para Spectrum 48 K

Colección tus Juegos Single
20 estuches de lujo con un programa para Spectrum 48 K incluyendo manual en castellano.

Colección tus juegos 4
5 estuches de lujo con 4 programas cada uno incluyendo manual de instrucciones en castellano para Spectrum 48 K.

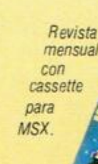
Revista con cassette de juegos de aparición mensual para Spectrum 48 K.

Seis cassettes con programas estrellas presentados en estuche de lujo para Spectrum 48 K
P.V.P. 1.795 pts.
Catálogo con instrucciones en castellano.



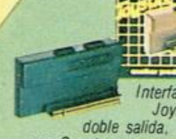
MONSER, S.A.
C/ Argos, 9
28037 Madrid
Tlf. 742 72 12 / 96





Revista mensual con programas listados para Spectrum, M S X, Amstrad y Commodore, incluyendo cinta virgen.

Revista mensual con cassette para MSX.



Interface para Joystick de doble salida, especial para Spectrum.



Lápiz de Luz para Spectrum.



Joystick Quinckshot I





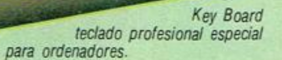
Cassette virgen C-30, especial para ordenadores.

Dos programas de juegos para Spectrum 48 K, con cinta virgen de regalo.



Joystick Quickshot II





Key Board
teclado profesional especial para ordenadores.



Joystick «Challenger»

Nota: Todos los cassettes llevan los programas grabados en ambas caras.



AÑO I - N.º 7 - 1985

DIRECTOR
José Nieto Rubio

COORDINADOR
Félix Santamaría Avila

SUPERVISOR SOFTWARE
Gustavo Cano Muñoz

DISEÑO
Angélica Arce

DIBUJOS
Ozzy

REDACCION
Juncal Feijoo
María Amaya
Belén Sánchez Vicente
Juan Antonio García

COLABORADORES
Mario Alvarez
Javier González
Agustín Barcos

PORTADA
Mauro Novoa

EDITA
MONSER, S. A.

DIRECTOR EDITORIAL
J. L. Cano Regidor

REDACCION, ADMINISTRACION Y PUBLICIDAD
Argos, 9
28037-MADRID
Tel. 742 72 12/96

PUBLICIDAD Y SUSCRIPTORES
Yolanda Bardillo

FOTOCOMPOSICION
CRISOL, S. A.

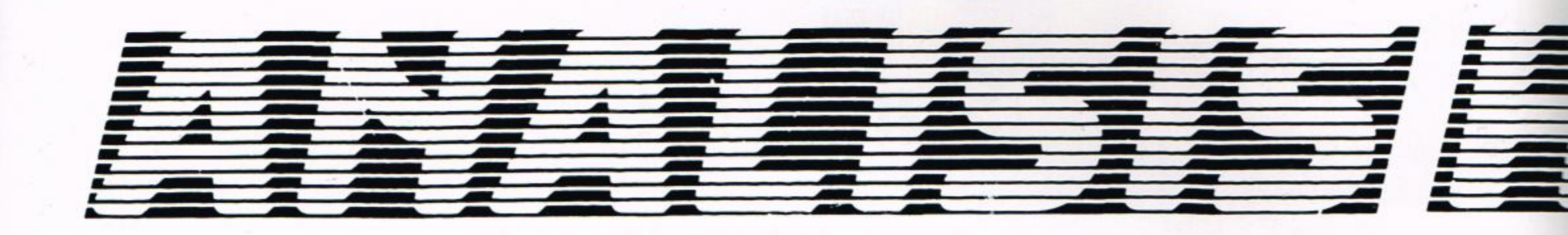
FOTOMECANICA
IMAGEN

IMPRIME
GRAFICAS IBARRA

DISTRIBUCION
DISPRENSA
Eduardo Torroja, 9

Depósito legal: M-10.328-1985

Sumario:



LA CINTA COMO SOPORTE DE INFORMACION (y II)

1.—Constitución de la cinta magnética

La cinta magnética es el medio en el que se almacena la información, por lo tanto, la calidad del sistema depende en gran parte de las características de la cinta.

La cinta, físicamente, está compuesta por una capa magnética distribuida uniformemente sobre un soporte plástico.

La capa magnética, que es la que está en contacto directo con las cabezas del magnetófono, se denomina pista o track. La misión del soporte plástico es proporcionar la sustentación física del material magnético.

El soporte debe tener una serie de propiedades físicas como son la resistencia a la tracción y al desgarro, flexibilidad mecánica, superficie lisa, poca dilatación con la temperatura, etc.

Es en la capa magnética donde se realiza el registro de la información propiamente dicho. Está formado por una dispersión uniforme de material ferromagnético en un fijador. Este material ferromagnético debe tener dos propiedades fundamentales: alta remanencia y alta retentividad.

Lo primero es necesario para que la relación señal/ruido sea lo suficientemente alta, es decir, cuanto más grande sea la remanencia, más grande será la señal máxima que se obtiene en reproducción.

La alta retentividad es necesaria para que las grabaciones no se vayan borrando progresivamente con el tiempo.

En resumen, debido a las propiedades de estos materiales, si aplicamos un campo H durante un corto intervalo de tiempo, comprendido entre cero y su valor de saturación, conseguimos en el material un campo remanente B proporcional al campo H aplicado, pudiendo de esta manera almacenar información.

2.—Tipos de compact cassette existentes en el mercado

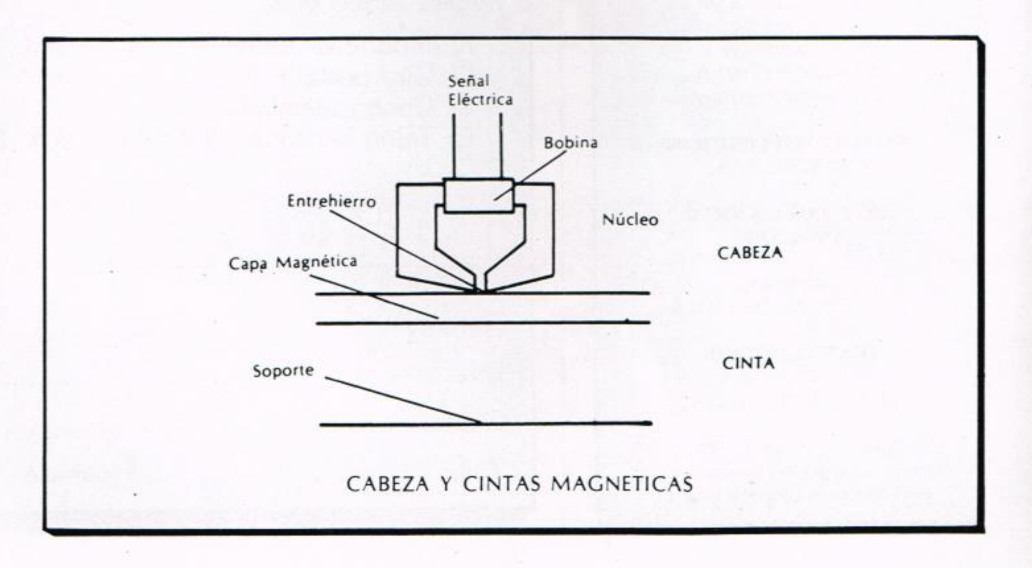
La clasificación la vamos a hacer respecto al compuesto ferromagnético existente en la cinta.

Vamos a dar dos características técnicas de cada tipo de cintas: la relación señal/ruido y la gama de frecuencias que se pueden reproducir con ese tipo de cinta.

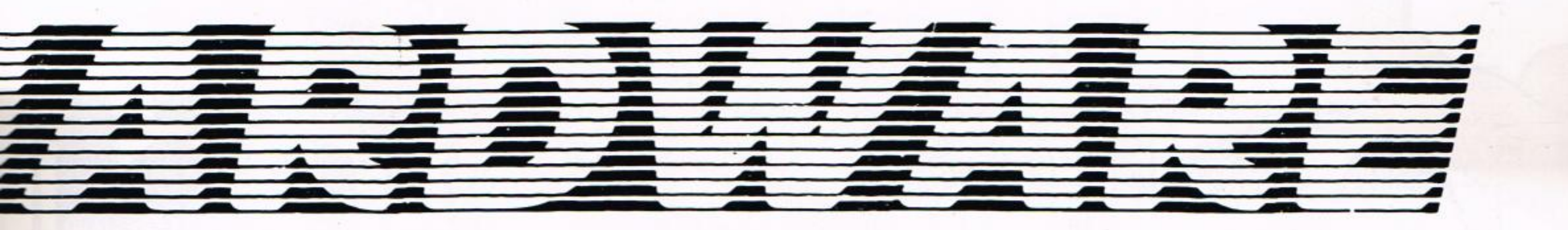
Lógicamente, los resultados obtenidos son para el mismo tipo de soporte plástico y para la misma cabeza de grabación-reproducción.

La relación señal/ruido es el cociente entre la señal pura propiamente dicha y el ruido que se superpone a esta señal, medido este cociente en decibelios (dB). Cuanto más grande sea la relación señal/ruido, mejor calidad tendrá la cinta.

La gama de frecuencias a reproducir va desde las frecuencias bajas (graves) a las frecuencias altas (agudos). Cuanto más amplia sea esta gama de frecuencias, mejor calidad tendrá la cinta.



CABEZA Y CINTAS MAGNETICAS



Veamos entonces cuáles son y cómo se denominan comercialmente, los distintos tipos de cinta:

Hierro normal

Relación señal/ruido .... 36 a 40 dB
Gama de frecuencias .... 50 a 5.500 Hz
El material utilizado es el óxido de hierro ($Fe_2 O_3$).

Hierro LH (L = Low noise, bajo ruido. H = High output, alta salida)

Relación señal/ruido .... 51 a 56 dB
Gama de frecuencias .... 40 a 9.000 Hz
El material utilizado es también óxido de hierro.

Hierro LH SUPER

Relación señal/ruido .... 54 a 59 dB
Gama de frecuencias .... 40 a 13.000 Hz
El material utilizado, al igual que en las dos anteriores, es el óxido de hierro.

Cromo

Relación señal/ruido .... 60 a 61,5 dB
Gama de frecuencias .... 30 a 15.000 Hz
El material utilizado es el óxido de cromo ($Cr O_2$).
El magnetófono debe de estar preparado para este tipo de cinta (posición del selector tipo de cinta $Cr O_2$).

Cobalto

Tiene las mismas características que el Cromo.
El material utilizado es el óxido de cobalto ($Co O_2$).

Ferrocromo

Relación señal/ruido .... 62 a 65,5 dB
Gama de frecuencias .... 25 a 16.000 Hz
El material utilizado es una capa de óxido de hierro en contacto con el soporte y otra capa de óxido de cromo sobre la anterior.
El magnetófono debe estar preparado para este tipo de cinta (posición del selector tipo de cinta $Cr O_2$).

Ferrocobalto

Tiene las mismas características que el ferrocromo.

Metal

Relación señal/ruido .... 68 a 72 dB
Gama de frecuencias .... 20 a 20.000 Hz
El material utilizado es hierro puro sin oxidar, recubierto por una película plástica muy fina para evitar que el hierro se oxide al entrar en contacto con el aire.
El magnetófono debe estar preparado para este tipo de cinta (posición del selector tipo de cinta METAL).

3.—Ordenador y cinta magnética, criterios de elección

Debido a las características del tratamiento y de la transmisión de datos de forma digital, no se necesita una alta fidelidad a la hora de reproducir una grabación digital para recuperar un programa anteriormente salvado, la cual sí se desearía al reproducir una grabación analógica musical.

Así, generalmente, se utilizan magnetófonos a cassette típicos para ordenadores que, funcionando perfectamente para señales digitales, no dan alta fidelidad en grabaciones musicales. La gama de respuesta en frecuencias de estos magnetófonos es de 40 a 10.000 Hz aproximadamente, eso significa que, utilizar cintas de cromo o de metal para grabaciones digitales es desperdiciar una serie de características, (y por supuesto de dinero), ya que este tipo de cintas ha sido diseñado para grabaciones musicales de alta fidelidad.

En cambio, algo que hay que exigir a una cinta que almacene programas, es un buen soporte plástico y una distribución uniforme de la capa ferromagnética que no dé altibajos que nos hagan perder algún dato que nos haría, a su vez, perder el programa.

En resumen, un tipo de cinta que tiene una buena relación calidad/precio, y que además, cumple ampliamente con las características necesarias para el almacenamiento de programas de ordenador, son las cintas Hierro LH SUPER, que además, tienen la ventaja de que cualquier magnetófono del mercado esté preparado para este tipo de cinta.

CODIGO MAQUINA

CAPITULO VII
(Continuación)

11.—Experimento (Continuación)

Como en la anterior entrega habíamos prometido, en esta vamos a dedicarnos a comentar la rutina que denominábamos TECLA y cuyo listado en Assembler incluíamos. En esta ocasión vamos a repetirla, pero junto a ella vamos a incluir el Código Máquina que le corresponde a cada instrucción. Para cumplimentar las direcciones que aparecen en la misma hemos considerado que esta rutina se carga en la dirección 40000.

Pasemos ahora a ver cómo se comporta en cada momento, y por lo tanto, qué podemos esperar de ella.

En primer lugar, abrimos el canal correspondiente al teclado, que es por donde nos van a venir los datos. Para ello se utilizan las instrucciones LD A,1 y CALL # 1601 que ya conocemos de la rutina de PRINT de capítulos anteriores.

A continuación, y una vez abierto el canal 1, correspondiente al teclado, se pasa a llamar a la rutina de la ROM que hace la función/INKEY$.

```
Direccion   Codigo Maquina   Assembler

 40000      3E 01            TECLA LD   A,1
 40002      CD 01 16               CALL #1601
 40005      CD 8E 02               CALL #028E
 40008      0E 00                  LD   C,0
 40010      C2 59 9C               JP   NZ,TECLN
 40013      CD 1E 03               CALL #031E
 40016      D2 59 9C               JP   NC,TECLN
 40019      15                     DEC  D
 40020      5F                     LD   E,A
 40021      CD 33 03               CALL #0333
 40024      C9                     RET
 40025      3E 00            TECLN LD   A,0
 40027      C9                     RET
```

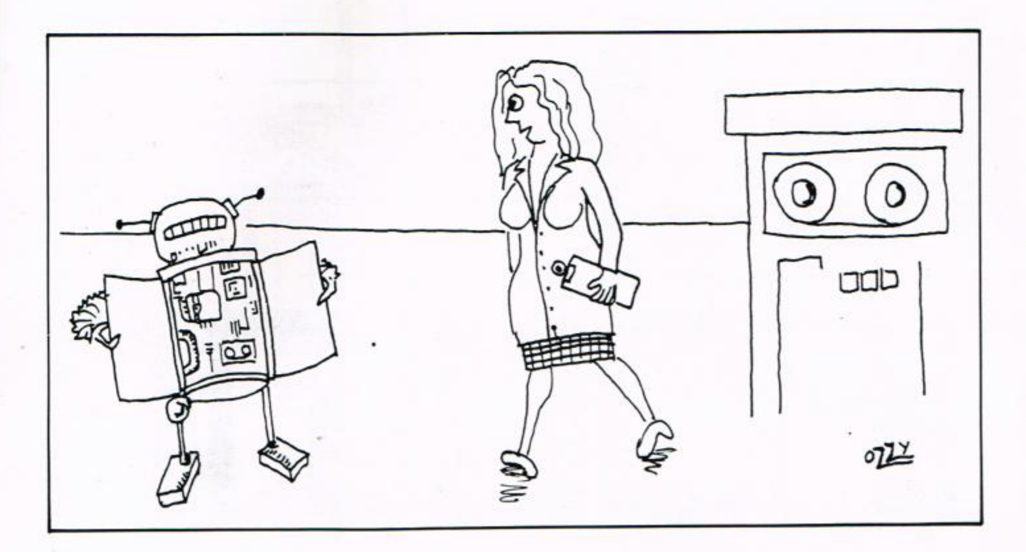
Es decir, analiza el teclado por si ha sido pulsado. Esta rutina se encuentra en la dirección de memoria # Ø28E. Tras la conexión con ella y cuando nos devuelve el control nos encontramos con las siguientes consecuencias.

— El registro E contiene un valor hexadecimal entre # ØØ y # 27 si se ha pulsado alguna tecla, o en caso contrario el valor # FF.

— El registro D contiene una indicación de cuál de las teclas simples (CAPS SHIFT o SYMBOL SHIFT) ha sido pulsada, y si no hubiéramos pulsado ninguna de las dos pues también contendrá el valor # FF.

— El flag de cero (Z) es desactivado si son pulsadas dos teclas, siempre y cuando una de ellas no sea alguna de las que hemos denominado simples (CAPS SHIFT o SYMBOL SHIFT).

La siguiente instrucción, LD, C, Ø, la incluimos aquí para asegurarnos que en las posteriores rutinas de la



ROM, que van a ser llamadas, no se consideren más que los modos de cursor K, L o C. Digamos que constituyen una operación complementaria que no forma parte de la idea principal, pero que es conveniente incluirla.

La instrucción JP NZ, TECLN, que va a continuación hace que el control del programa se transfiera a la instrucción nombrada como TECLN en el caso de que se pulsen más de una tecla, pues, como dijimos al comentar la anterior rutina de la ROM, cuando se pulsan dos teclas se desactiva el flag de cero (Z).

Si se ha pulsado una tecla, el control del programa continúa en secuencia y, por lo tanto, pasa el control a la rutina de la ROM, que se halla situada en la dirección # Ø31E, por medio de la instrucción CALL # Ø 31 E. Esta rutina comprueba que es una de las teclas aceptadas, según el modo de cursor consideraro (he aquí el valor de la instrucción anteriormente mencionada). Además, comprueba que no sea únicamente una de las denominadas «simples». En el caso de que no encuentre una tecla reconocible, o es únicamente una tecla simple la pulsada, deja desactivado el flag de acarreo (C), también llamado «carry flag».

La instrucción JP NC, TECLN considera tal circunstancia, y, si ese flag está desactivado, cede el control a la instrucción llamada TECLN.

Con la instrucción DEC D buscamos efectos parecidos a los ya comentados cuando cargábamos en el registro C un cero (LD C,Ø) pero en esta ocasión indicamos a la siguiente rutina de la ROM que los tipos de cursor a emplear son los modos «L» y «C» (minúsculas y mayúsculas, respectivamente).

La instrucción LD E,A es una preparación para conectarnos con la última rutina de la ROM que utilizamos en TECLA. La rutina # Ø333 a la que entramos por medio de la instrucción CALL # Ø 333, necesita que al entrar, el número de la tecla se encuentre en el registro E. Pero dicho número ha sido puesto en el registro A durante la ejecución de la rutina anterior (# Ø 31 E).

Cuando la rutina situada en # Ø 333 nos devuelve el control, nos encontramos en el registro A el código ASCII correspondiente a la tecla pulsada. Sólo queda retornar, mediante RET, al programa que llamó a esta rutina TECLA, y con ello se finaliza la ejecución de la rama principal de la misma.

Solamente nos queda describir la rama que denominamos TECLN. A esta parte se bifurca únicamente cuando se dan las circunstancias ya apuntadas anteriormente de que no hay pulsada ninguna tecla, o que se pulsan simultáneamente dos. No creo que sea necesario ser muy exhaustivo para explicar qué es lo que sucede cuando se ejecutan las dos instrucciones que la forman. En efecto, la LD A,Ø hace que en el registro Acumulador se cargue un valor Ø, y la siguiente instrucción, RET, hace que se transfiera el control a la instrucción que sigue en secuencia a la que llamó a la rutina.

Esta rutina puede sufrir variaciones, y de hecho según vayamos conociendo más instrucciones iremos modificándola, pero nos vale en su estado inicial para investigar si en el teclado está pulsada alguna tecla. Si al salir de la misma encontramos que en el registro A hay un cero binario, nos indicará que no hay tecla pulsada, o se pulsan simultáneamente dos.

Dejemos que aquellos lectores con mayores conocimientos realicen los cambios que consideren más oportunos y nosotros vamos a plantearnos un programita que haga uso de esta rutina que acabamos de comentar.

Lo primero que uno se plantea al hacer un programa es su función, su finalidad. En este caso, la finalidad del programa es puramente didáctica, como es evidente, pero el programa tiene que hacer algo, y esto es lo que hay que definir.

Vamos a realizar un programa que sea muy sencillo y que consista en la incorporación de cualquier letra, introducida a través de teclado, a un mensaje que aparecerá en pantalla.

Aunque sea salvando muchas distancias, vamos a llevar a cabo una parte ínfima de un procesador de textos. Claro que con las pocas instrucciones que conocemos no podemos hacer grandes cosas todavía.

Para mandar terminar la ejecución del programa y que nos devuelva el control al Sistema Operativo, o Basic, pondremos la condición de que eso se produzca cuando se pulse la tecla ENTER.

Una vez definido el problema vamos a dividirlo en grandes bloques que contemplen las diferentes funciones a realizar.

En principio se pueden apreciar tres fases diferentes:

a. Leer el teclado.

b. Incorporar la tecla leída al *mensaje*.

c. Escribir el mensaje.

Tanto la función *a* como la *c* están perfectamente definidas, e incluso programadas. Por lo tanto los condicionantes referidos a la comunicación con ellas ya están determinados.

Antes de comenzar a programar la fase b, es conveniente hacer un pequeño estudio de cuál va a ser el formato del área en donde vamos a almacenar las letras que son introducidas a través del teclado, y que van a formar el mensaje. Esto nos llevará a considerar también la forma en que debe relizarse el almacenamiento de las letras.

Como es lógico, aunque no preceptivo, el mensaje vamos a exhibirlo siempre comenzando en la posición (Ø Ø) de la pantalla. Esto se consigue de dos modos distintos: primero, borrando el área de pantalla mediante CLS y segundo, obligándola a través de la opción AT de la PRINT. Nosotros, para ver alguna cosa más, nueva, vamos a hacerlo de las dos formas. De este modo, cualquier lector podrá hacerlo según su gusto.

Pues bien, como ya vimos en anteriores experimentos, el mensaje debe contener los códigos correspondientes a los caracteres de control que se deseen utilizar, en este caso el AT (Código # 16). Detrás de el código AT deben figurar las coordenadas del punto donde se desea que comience el mensaje, en este caso Ø, Ø. De este modo ya tenemos la estructura embrionaria del mensaje.

Mensaje:	16	00	00

Dado que para la utilización de la rutina PRINT necesitamos conocer la longitud y ésta irá aumentando en una unidad por cada letra que incorporemos a través del teclado, sería conveniente guardar la longitud en algún sitio. Lo vamos a poner en los dos bytes anteriores al mensaje. También voy a guardar la dirección en donde tengo que incorporar la siguiente letra para no equivocarme. Pues ahora la estructura del área puede ser la siguiente:

Dirección letra:			
Longitud Mensaje:			
Mensaje:	16	00	00

Esos campos deben contener siempre un valor adecuado. Por ejemplo, la longitud del mensaje inicialmente debe ser igual a 3 y la dirección de la primera letra a poner en el mensaje deberá ser la que corresponde a la que sigue a lo ya conocido.

Para llevar a cabo todas estas operaciones iniciales se me ocurre la idea de crear otra función específica que las reúna y que, por hacerse al principio del programa, vamos a denominar de «Inicialización». De este modo, todo el programa podemos describirlo como un pequeño conjunto de funciones que se realizan secuencialmente, y lo representamos mediante lo que denominaremos «diagrama, u organigrama de bloques» como el representado por la figura 1. Cada una de las funciones de que consta dicho organigrama será llamada por medio de una instrucción CALL que ya conocemos.

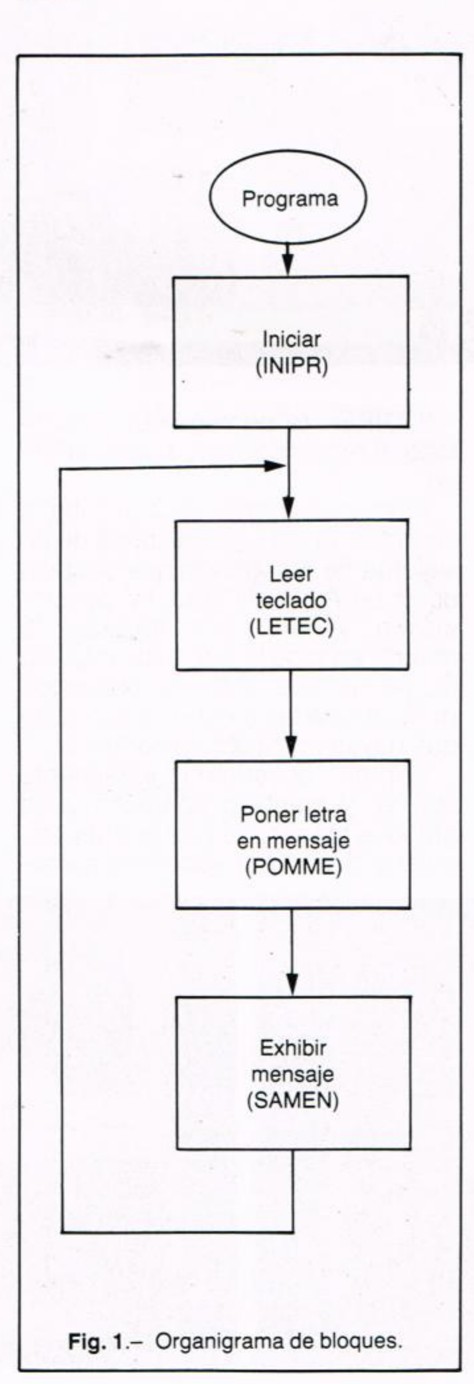


Fig. 1.– Organigrama de bloques.

Antes de continuar con más elucubraciones frente al programa, hemos de hacer un pequeño alto para señalar que las instrucciones de Assembler están compuestas de tres partes: nombre simbólico, umónico o mnemotécnico y parámetros. El nombre simbólico es el equivalente al número de instrucción en Basic, con la diferencia de que en Assembler sólo se nombra a aquellas instrucciones a las que nos interesa nombrar, y en Basic hay que numerar todas o casi todas. En realidad no damos un nombre a la instrucción sino a la dirección en que se encuentra esa instrucción.

El umónico, o mnemotécnico, es el código de la operación que debe ejecutarse. Equivale en Basic a las instrucciones ya conocidas como GOTO, INPUT, LET, etc. En Assembler ya conocemos algunos de estos códigos de operación, como CALL, LD, JP, etc.

Por último, los parámetros son aquellos elementos entre los que va a realizarse las operaciones descritas por el umónico. Por ejemplo, en la instrucción LD A,B los parámetros serían los registros A y B. En ocasiones, los parámetros no se explicitan en la instrucción sino que ya figuran en ella de un modo implícito. Más adelante tendremos varios ejemplos de este tipo de parámetros implícitos.

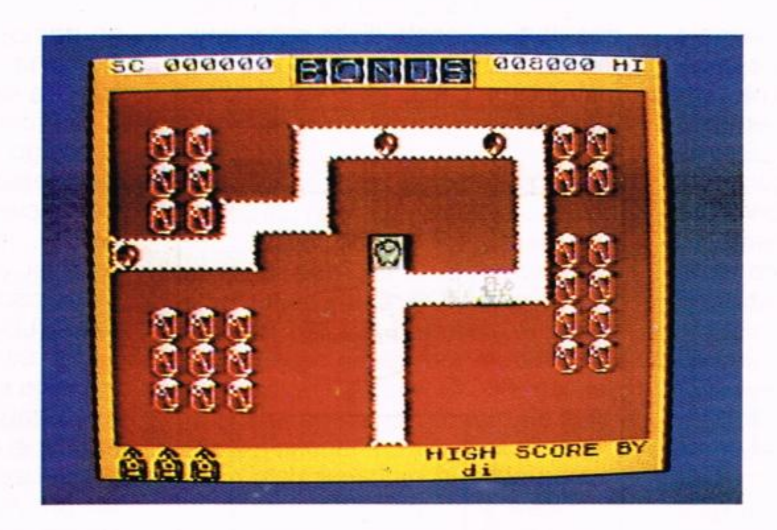
Retomemos el hilo de nuestro programa después de dejar satisfecho, así lo espero, a alguno de los lectores que andaba preocupado por lo anterior.

Una vez dividido el programa en funciones, o bloques, vamos a describir y programar cada una de ellas, del modo y manera que al final podamos descubrir lo fácil que es hacer un programa muy complicado cuando lo dividimos en problemitas chiquititos. Como se puede observar en la figura 1, bajo la descripción de cada uno de los bloques hemos puesto el nombre simbólico. Ese es el que utilizaremos a la hora de programar. También nos falta el nombre simbólico de la estructura, mejor dicho, de los campos de la estructura. Llamaremos DIRLE a la dirección de la letra a poner, LONME a la longitud del mensaje y MENSA a la dirección del mensaje.

Sentadas estas premisas vamos a describir una a una las diversas funciones:

(Continúa en pág. 12)

QUESO DURO



En este precioso juego, tendrás que dirigir a Fredic el granjero, en su excavadora, para limpiar de barriles el campo, y recoger los trozos de queso que aparecerán de cuando en cuando en el centro de la pantalla.

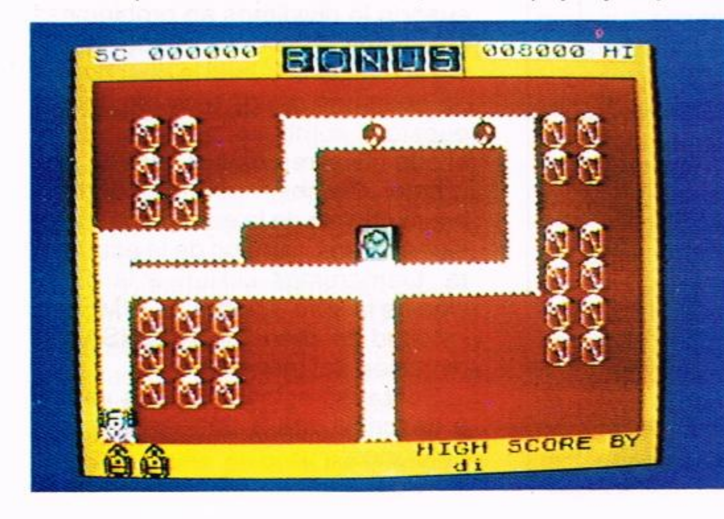
Pero eso no es todo, sería demasiado fácil.

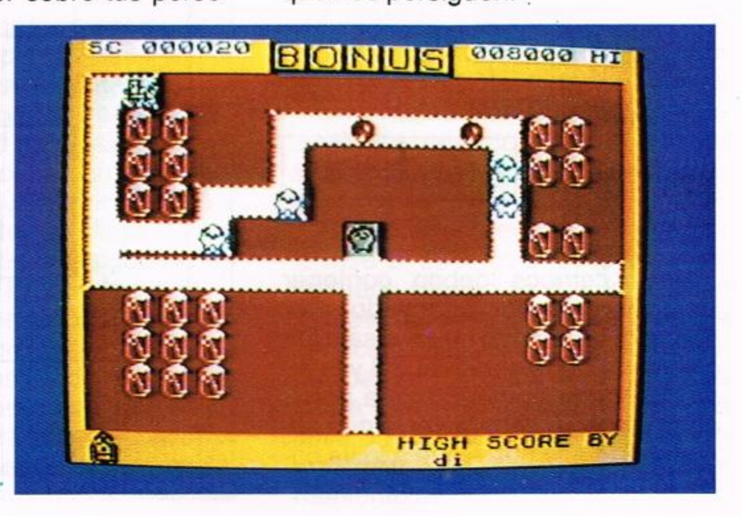
Para estropearte el día de trabajo, existen unos apestosos animalitos que no debes en ningún momento perder de vista; empieza cuanto antes a quitar barriles sin esperar a que salgan y eso que llevarás adelantado, pues una vez en acción, los «BUGIES» te perseguirán durante todo el recorrido para acabar contigo.

Afortunadamente todo no iba a ser malo, tu vehículo dispone de un sistema de fumigación que despide dosis de AGROACIDO, un peligrosísimo veneno que produce la muerte en todo lo que toca, incluido tú, por lo que una vez disparado (mediante la tecla «M»), te aconsejo que huyas en dirección contraria.

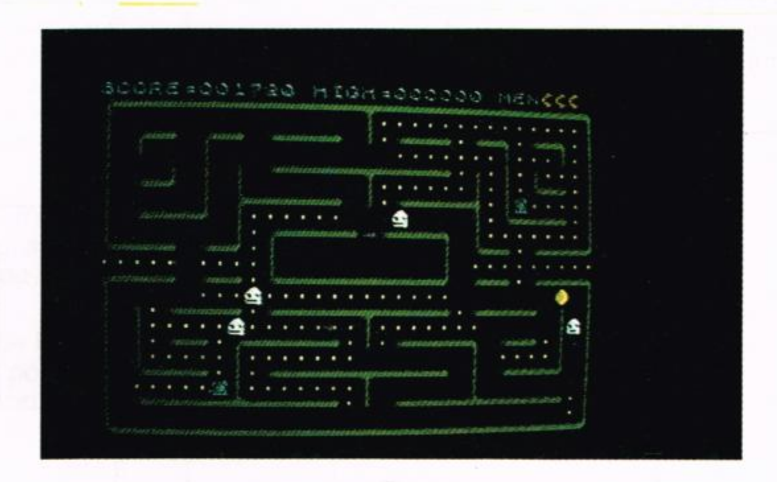
Habrás observado igualmente que en la pantalla aparecen unas grandes manzanas que podrás empujar y dejar caer sobre tus perseguidores, si les das los eliminas y si no, la manzana impedirá el paso de los terribles «BUGIES».

Como ya habrás visto, el juego consiste en recoger los barriles de toda la pantalla, comerte el queso del centro y acabar con todos los «BUGIES» que no te dejan ni a sol ni a sombra, por todos los medios a tu alcance, manzanas y agroácidos, para una vez que lo hayas conseguido pasar a la siguiente pantalla, que como es lógico será algo más difícil de conseguir escapar y acabar con todos los extraños bichitos que nos persiguen.

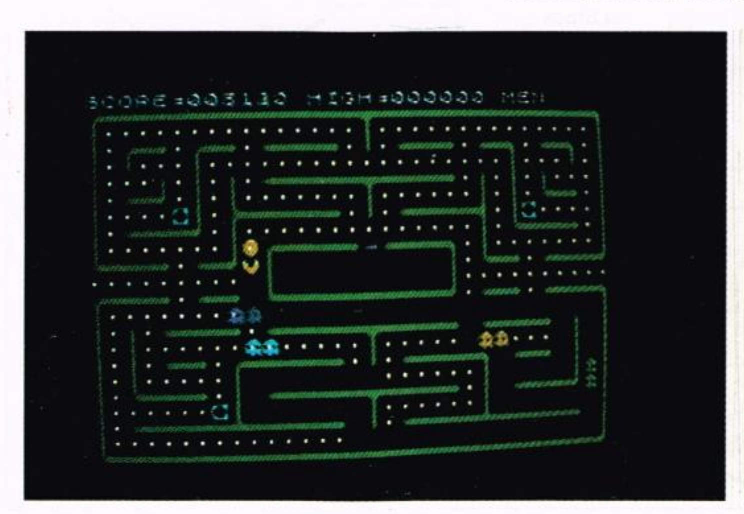


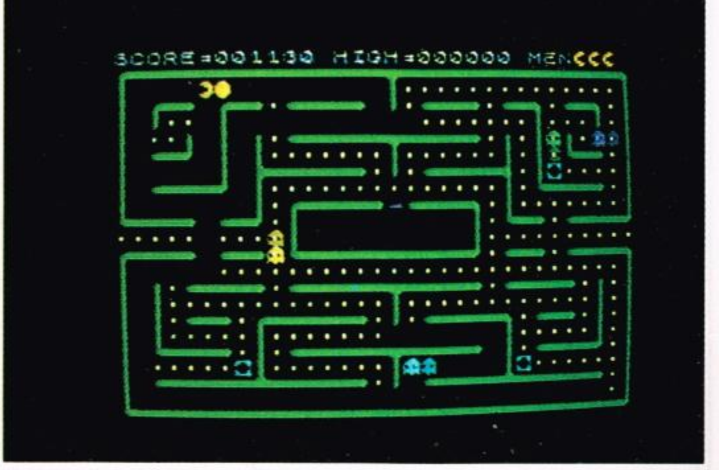


COME-COME



En cuanto a nuestro segundo juego, ¿qué os vamos a decir?, ¿quién no ha jugado alguna vez con el clásico comecocos?, este programa ya es tan conocido como la propia electrónica de consumo y lógicamente en este especial no podía faltar este juego, que no por conocido es menos agradable, y que no todos los usuarios de Spectrum tienen en su programoteca, o al menos no poseen una versión tan conseguida del mismo, con el que esperamos paséis tan buenos momentos como nosotros los hemos pasado a la hora de seleccionarlo entre montones de ellos.





Inicialización (INIPR)

Como ya hemos dicho antes, esta función está destinada a inicializar los valores que deben contener los campos que hemos descrito (DIRLE, LONME).

Principalmente consiste en la carga de registros pares con los valores de inicio y a continuación se salvan, o almacenan, en los campos correspondientes.

Esta función se realiza una sola vez, cuando se entra en el programa. Su organigrama y codificación es el representado en la figura 2.

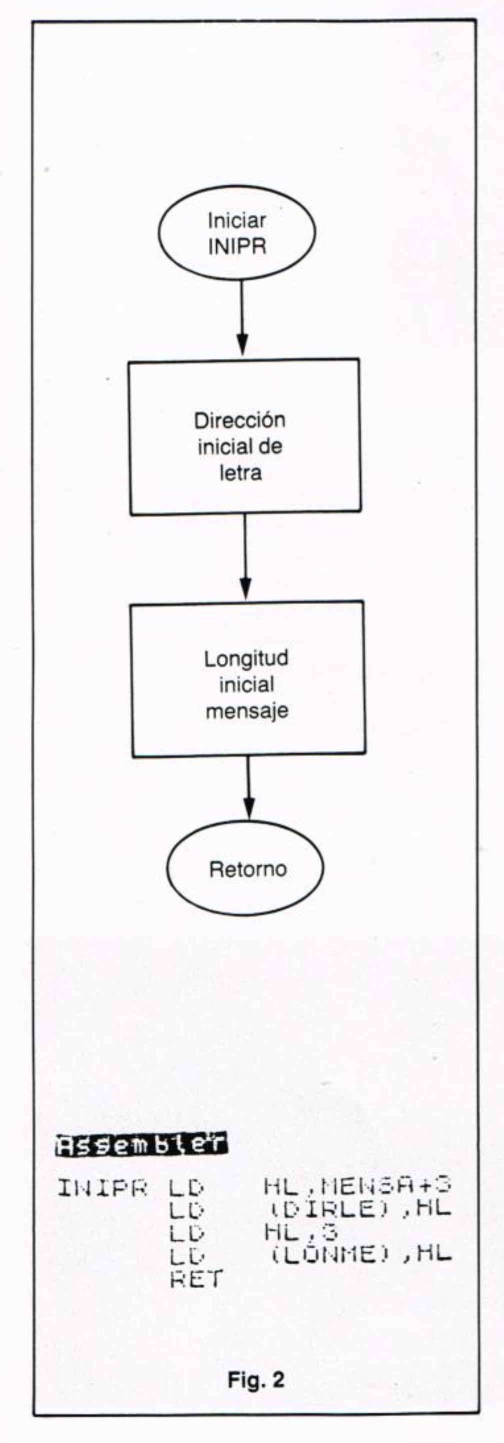


Fig. 2

Leer teclado (LETEC)

Este bloque es el encargado de llamar a la rutina TECLA que ya conocemos y, tras analizar el contenido del registo A que devuelve dicha rutina, tomar una decisión según las siguientes premisas:

— Si es un cero (# ØØ) vuelve otra vez a la rutina TECLA.
— Si es un trece (# ØD), código de la tecla ENTER, devuelve el control al Sistema Operativo, o programa Basic.
— Si es otro código distinto de los dos anteriores devuelve el control al programa principal escrito en Assembler.

Como en la rutina, o bloque, anterior también adjuntamos el organigrama y su codificación en Assembler.

Assembler

```
LETEC CALL TECLA
      SUB  13
      JP   NC,LETEC
      JP   Z,SABAS
      ADD  13
      RET
SABAS INC  SP
      INC  SP
      RET
```

Organigrama

Llamada a TECLA

Restar 13 al reg. A.

NO

Flag C

SI

NO

Flag Z

SI

Sumar 2 al Stack

Retorno

NO

Sumar 13 al ACUMULA.

RETORNO

Fig. 3

Sospecho que más de un lector se quedará con las ganas de saber todo lo que se ha hecho en esta rutinita. No es mi intención aburrir a las vacas con vulgaridades, pero tampoco quisiera que, por ser muy ligero, alguno se quedase a dos velas. Explicaré esta rutina con un poco de detalle.

La primera instrucción, la de llamar a la rutina TECLA, no creo que necesite explicación. Al salir de esta rutina el registro A contiene, o un cero binario, o un código superior al 32 que es el correspondiente al «espacio» en código ASCII. Si ahora le restamos 13 ¿qué es lo que puede pasar? Analicemos la situación:

— Si A tenía un cero, al restar 13 nos pondrá el carri-flag, o indicador de acarreo, activado.

— Si A tenía el código de la tecla ENTER, el trece, nos pondrá el flag de cero (Z), activado.

— Si A tiene un número superior a 13 no activará ni el carry-flag ni el flag de cero (Z).

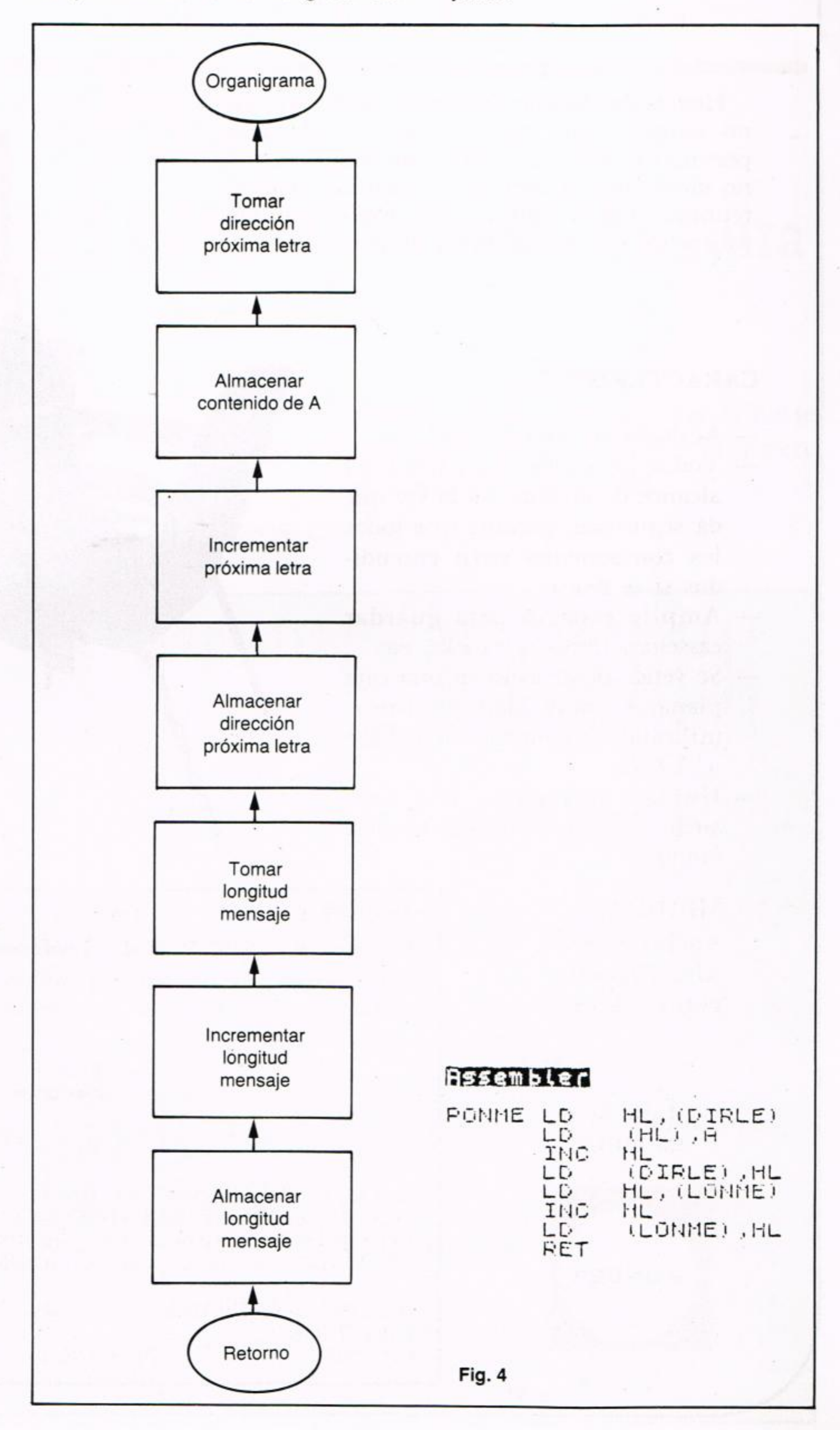
Pues bien, las dos instrucciones siguientes se aprovechan de tal circunstancia y bifurcan a sitios diferentes según el flag activado. Si ninguno de los dos casos anteriores se ha producido, vuelven a sumarse los 13 que habíamos restado al Acumulador, con lo que restauramos el valor inicial de A y por último retornamos.

Por último nos queda considerar la rutina SABAS que hace que retornemos al Basic. ¿Cómo? Pues muy sencillo. Cuando se conecta el Basic con el programa principal en C/M, se deja en el stack la dirección de vuelta. La dirección que apunta el registro SP es la que contiene esa dirección de vuelta. Cuando hacemos una llamada CALL, entre otras cosas, el registro SP se *decrementa* en dos unidades y ahí se coloca la dirección de vuelta. Cuando entramos en esta rutina, el stack apunta a la dirección de vuelta, al programa principal. Si ahora nosotros *incrementamos* en dos unidades el contenido del registro SP, este direccionará la dirección de vuelta al BASIC. Por lo tanto, la instrucción RET no nos llevará al programa principal sino al BASIC.

Poner tecla en mensaje (POMME)

Este bloque tiene por función la de incorporar la letra correspondiente a la tecla pulsada en el mensaje, situándola en el lugar oportuno, y actualizando los valores correspondientes a la longitud del mensaje y a la dirección en donde debe colocarse la siguiente letra.

Considero que esta función no necesita de explicación alguna una vez que se ven con claridad todos los pasos realizados, mediante el organigrama y la codificación adjunta.




```
PONME LD    HL,(DIRLE)
      LD    (HL),A
      INC   HL
      LD    (DIRLE),HL
      LD    HL,(LONME)
      INC   HL
      LD    (LONME),HL
      RET
```

Fig. 4

ORDENA TU ORDENADOR

Quítale Trabajo a tu Micro

P.V.P. sólo 8.975

Hemos diseñado la estantería ideal para que no tengas tirado por la casa tu ordenador personal y accesorios. Con este complemento no molestarás al resto de tu familia y tendrás reunido todo tu equipo, sacándole el máximo provecho, sin que nadie te moleste.

CARACTERISTICAS

- Acabado en efecto roble.
- Todos los cables están fuera del alcance de la vista y a la vez que dá seguridad, permite que todos los componentes estén encendidos si se desea.
- Amplio espacio para guardar cassettes, libros, joysticks, etc.
- Se vende desarmado en una caja plana, es muy fácil de armar, utilizando solamente una llave ALLEN.
- Unidad de puente: 56,5 cms. ancho. 17 cms. alto. 30,48 cms. fondo.

MEDIDAS
Ancho 83,5 cm.
Alto 79,5 cm.
Fondo 60 cm.

Con la garantía



MONSER S. A.

C/ Argos, 9 - 28037 Madrid . Teléfonos: (91) 742 72 12 - 742 72 96

Por favor envíenme los siguientes gabinetes:

REF. No.	CANTIDAD	PRECIO	TOTAL
....	....	8.975 C.U. Ptas	Ptas. ....

Mas gastos de envío ....

TOTAL PTAS. ....

□ TALON ADJUNTO □ TALON CONFORMADO ADJUNTO □ GIRO POSTAL □ GIRO TELEGRAFICO □ CONTRA REEMBOLSO □ TRANSFERENCIA BANCARIA □ (Cta. No. 836940 del Bco. Central). □ PAGO APLAZADO - SOLICITE INFORMACION.

NOMBRE Y APELLIDOS ....
DIRECCION ....
CIUDAD .... PROVINCIA .... TEL ....



Ganador concurso 48K n.º 1

Luis Miguel López y Antonio Anguita

El programa «TT64C * 2TP» como véis son dos listados, el primero sirve para crear una línea 0 en la que se encuentra la rutina en código máquina de las 64 columnas; una vez tecleado el listado, al ejecutarlo aparece una línea cero, entonces hay que borrar el resto del programa (líneas restantes hasta la 1340).

El segundo listado es propiamente el programa «TT64C *25p».

```
  10 FOR N=23755 TO 25065
  20 READ D
  30 POKE N,D
  40 NEXT N
  50 DATA 0,0,20,5,234,246,92,19
6,21,0
  50 DATA 175,50,212,92,33,27,0.
34,30,92
  60 DATA 33,246,92,34,208,92,33
,195,21,34
  70 DATA 210,92,253,54,71,0,33,
0,0,34
  80 DATA 176,92,201,8,253,126,7
1,254,0,40
  90 DATA 30,254,2,40,12,253,53,
71,8,254
 100 DATA 64,48,16,253,119,119,2
01,253,53,71
 110 DATA 8,254,24,48,4,253,119,
118,201,207
 120 DATA 4,8,254,32,48,16,254,1
3,202,170
 130 DATA 93,254,22,32,5,253,54,
71,2,201
 140 DATA 62,63,254,165,56,2,62,
63,8,253
 150 DATA 126,118,254,24,56,8,20
5,254,13,62
 160 DATA 23,253,119,118,253,78,
119,203,57,87
 170 DATA 15,15,15,230,224,111,1
22,230,24,246
 180 DATA 64,103,6,0,9,8,235,38,
0,111
 190 DATA 41,41,41,37,1,182,93,9
,235,253
 200 DATA 203,119,70,32,19,6,8,2
6,203,22
 210 DATA 203,22,203,22,203,22,2
37,103,19,36
 220 DATA 16,241,24,17,6,8,26,20
3,30,203
 230 DATA 30,203,30,203,30,237,1
11,19,36,16
 240 DATA 241,37,124,15,15,15,23
0,3,246,88
 250 DATA 103,253,126,85,119,253
,126,119,60,254
 260 DATA 64,32,8,253,54,119,0,2
53,52,118
 270 DATA 201,253,119,119,201,0,
0,0,0,0
 280 DATA 0,0,0,0,2,2,2,2,0,2
 290 DATA 0,0,5,5,0,0,0,0,0,0
 300 DATA 5,7,5,5,7,5,0,0,2,7
 310 DATA 4,7,1,7,2,12,13,1,2,4
 320 DATA 8,11,3,0,2,5,2,6,11,6
 330 DATA 0,0,2,4,0,0,0,0,0,0
 340 DATA 2,4,4,4,4,2,0,0,4,2
 350 DATA 2,2,2,4,0,0,0,5,2,7
 360 DATA 2,5,0,0,0,2,2,7,2,2
 370 DATA 0,0,0,0,0,0,2,2,4,0
 380 DATA 0,0,0,7,0,0,0,0,0,0
 390 DATA 0,0,6,6,0,0,1,1,2,2
 400 DATA 4,4,0,0,2,5,5,5,5,2
 410 DATA 0,0,2,6,2,2,2,7,0,0
 420 DATA 2,5,1,2,4,7,0,0,6,1
 430 DATA 6,1,1,6,0,0,1,3,5,5
 440 DATA 7,1,0,0,7,4,6,1,1,6
 450 DATA 0,0,3,4,6,5,5,2,0,0
 460 DATA 7,1,2,2,4,4,0,0,2,5
 470 DATA 2,5,5,2,0,0,2,5,5,3
 480 DATA 1,6,0,0,0,0,2,0,0,2
 490 DATA 0,0,0,2,0,0,2,2,4,0
 500 DATA 0,1,2,4,2,1,0,0,0,0
 510 DATA 7,0,7,0,0,0,0,4,2,1
 520 DATA 2,4,0,0,2,5,1,2,0,2
 530 DATA 0,0,6,15,13,10,8,7,0,0
 540 DATA 2,5,5,7,5,5,0,0,6,5
 550 DATA 6,5,5,6,0,0,2,5,4,4
 560 DATA 5,2,0,0,6,5,5,5,5,6
 570 DATA 0,0,7,4,6,4,4,7,0,0
 580 DATA 7,4,7,4,4,4,0,0,2,5
 590 DATA 4,7,5,2,0,0,5,5,7,5
 600 DATA 5,5,0,0,7,2,2,2,2,7
 610 DATA 0,0,1,1,1,5,5,2,0,0
 620 DATA 5,5,6,5,5,5,0,0,4,4
 630 DATA 4,4,4,7,0,0,5,7,7,5
 640 DATA 5,5,0,0,6,5,5,5,5,5
 650 DATA 0,0,2,5,5,5,5,2,0,0
 660 DATA 6,5,5,6,4,4,0,0,2,5
 670 DATA 5,5,5,3,1,0,7,5,5,6
 680 DATA 5,5,0,0,3,4,2,1,1,6
 690 DATA 0,0,7,2,2,2,2,2,0,0
 700 DATA 5,5,5,5,5,7,0,0,5,5
 710 DATA 5,5,5,2,0,0,5,5,7,7
 720 DATA 7,2,0,0,5,5,2,2,5,5
 730 DATA 0,0,5,5,5,2,2,2,0,0
 740 DATA 7,1,2,2,4,7,0,0,7,4
 750 DATA 4,4,4,7,0,0,4,4,2,2
 760 DATA 1,1,0,0,7,1,1,1,1,7
 770 DATA 0,0,2,7,2,2,2,2,0,0
 780 DATA 0,0,0,0,0,0,15,0,2,5
 790 DATA 4,15,4,15,0,0,0,6,1,7
 800 DATA 5,7,0,0,4,4,6,5,5,6
 810 DATA 0,0,0,3,4,4,4,3,0,0
 820 DATA 1,1,3,5,5,3,0,0,0,2
 830 DATA 5,6,4,3,0,0,3,4,6,4
 840 DATA 4,4,0,0,0,3,5,5,3,1
 850 DATA 6,0,4,4,6,5,5,5,0,0
 860 DATA 2,0,6,2,2,7,0,0,1,0
 870 DATA 1,1,1,5,2,0,4,5,6,5
 880 DATA 5,5,0,0,4,4,4,4,4,3
 890 DATA 0,0,0,5,7,7,5,5,0,0
 900 DATA 0,6,5,5,5,5,0,0,0,2
 910 DATA 5,5,5,2,0,0,0,6,5,5
 920 DATA 6,4,4,0,0,3,5,5,3,1
 930 DATA 1,0,0,3,4,4,4,4,0,0
 940 DATA 0,3,4,2,1,6,0,0,2,7
 950 DATA 2,2,2,1,0,0,0,5,5,5
 960 DATA 5,3,0,0,0,5,5,5,5,2
 970 DATA 0,0,0,5,7,7,7,2,0,0
 980 DATA 0,5,5,2,5,5,0,0,0,5
 990 DATA 5,5,3,1,6,0,0,7,1,2
1000 DATA 4,7,0,0,3,2,4,2,2,3
1010 DATA 0,0,2,2,2,2,2,2,0,0
1020 DATA 6,2,1,2,2,6,0,0,5,10
1030 DATA 0,0,0,0,0,6,9,11,13,13
1040 DATA 11,9,6,0,0,0,0,0,0,0
1050 DATA 0,3,3,3,3,0,0,0,0,12
1060 DATA 12,12,12,0,0,0,0,15,15
,15
1070 DATA 15,0,0,0,0,0,0,0,0,3
1080 DATA 3,3,3,3,3,3,3,3,3,3
1090 DATA 3,12,12,12,12,3,3,3,3,
15
1100 DATA 15,15,15,3,3,3,3,0,0,0
1110 DATA 0,12,12,12,12,3,3,3,3,
12
1120 DATA 12,12,12,12,12,12,12,1
2,12,12
1130 DATA 12,15,15,15,15,12,12,1
2,12,0
1140 DATA 0,0,0,15,15,15,15,3,3,
3
1150 DATA 3,15,15,15,15,12,12,12
,12,15
1160 DATA 15,15,15,15,15,15,15,1
5,15,15
1170 DATA 15,6,9,9,15,9,9,9,0,14
1180 DATA 9,9,14,9,9,14,0,6,9,8
1190 DATA 8,8,9,6,0,14,9,9,9,9
1200 DATA 9,14,0,15,8,14,8,8,8,1
5
1210 DATA 0,15,8,8,14,8,8,8,0,6
1220 DATA 9,8,10,9,9,6,0,9,9,15
1230 DATA 9,9,9,9,0,14,4,4,4,4
1240 DATA 4,14,0,1,1,1,1,1,9,6
1250 DATA 0,9,10,12,10,10,9,9,0,
8
1260 DATA 8,8,8,8,8,15,0,9,9,15
1270 DATA 15,9,9,9,0,9,13,13,11,
11
1280 DATA 9,9,0,6,9,9,9,9,9,6
1290 DATA 0,14,9,9,14,8,8,8,0,6
1300 DATA 9,9,9,9,9,6,2,14,9,9,
1310 DATA 14,10,9,9,0,6,9,8,6,1
1320 DATA 9,6,0,15,2,2,2,2,2,2
1330 DATA 0,9,9,9,9,9,9,6,0,0
1340 DATA 195,144,234,13,0,1,103
,0,224,58
1350 DATA 224
```

```
  10 GO TO VAL "30"
  20 LET N$="": RANDOMIZE USR VA
L "23765": LET B$="
                   ": GO SUB VAL
 "1010"
  30 RANDOMIZE USR VAL "23765":
CLEAR VAL "65535": LET C=CODE ""
: LET A$="": LET B$="
                   ": GO SUB V
AL "9200": CLS : IF C=VAL "13" T
HEN GO TO CODE "7"
  40 DIM T$(P*VAL "21",CODE "@")
: DIM A(VAL "2"): LET L=VAL "1"
  55 PRINT BRIGHT VAL "1";AT COD
E "",VAL "3";"INTRODUZCA TEXTO O
 COMANDO": PRINT #CODE ""; BRIGH
T VAL "1";AT CODE "",CODE "";B$+
B$
  57 PRINT #0;AT 0,0; BRIGHT 1;
FLASH 1;">": FOR N=64 TO 1 STEP
-1: PRINT AT 0,0; FLASH 1; BRIGH
T 1;P*21-L;AT 0,32-LEN STR$ N; F
LASH 1; BRIGHT 1;N: IF LEN STR$
N=1 THEN PRINT ;AT 0,30; BRIGHT
1; FLASH 1;" "
  60 PAUSE 0: LET A$=CHR$ PEEK 2
3560: GO TO 30*(CODE A$=13 OR CO
DE A$=12 OR (A$="*" AND N=CODE "
@")) +5*(A$>=" " AND A$<="©" AND
NOT (A$="*" AND N=CODE "@")) +60
  65 LET T$(L,65-N)=A$: PRINT #0
; BRIGHT 1;AT 0,0;T$(L, TO 65-N)
;: IF N>VAL "1" THEN PRINT #CODE
 ""; BRIGHT VAL "1"; FLASH VAL "
1";">"
  66 IF N<VAL "5" THEN BEEP VAL
".01",CODE "#"
  70 NEXT N
  90 IF A$="*" AND N=CODE "@" TH
EN INPUT BRIGHT VAL "1";"INTRODU
ZCA COMANDO "; LINE A$: GO SUB C
ODE "x": GO TO CODE "7"
  91 IF (CODE A$=VAL "12" AND N=
CODE "@") OR A$="*" THEN GO TO C
ODE ";"
  92 IF CODE A$=VAL "12" THEN LE
T N=N+VAL "1": LET T$(L,CODE "A"
-N)=" ": PRINT #CODE ""; BRIGHT
VAL "1";AT CODE "",CODE "";T$(L,
 TO CODE "A"-N-VAL "1");: PRINT
#CODE ""; BRIGHT VAL "1"; FLASH
VAL "1";">"; FLASH CODE "";" ":
PRINT AT CODE "",VAL "32"-LEN ST
R$ N; BRIGHT VAL "1"; FLASH VAL
"1";N: GO TO CODE ";"
 100 PRINT #VAL "4";AT VAL "22",
CODE "";T$(L);: RANDOMIZE USR VA
L "3190": LET L=L+VAL "1": IF L>
P*VAL "21" THEN CLS : PRINT AT V
AL "8",VAL "2"; BRIGHT VAL "1";"
NO PUEDE INTRODUCIR MAS TEXTO";A
T VAL "10",VAL "3";"DIMENSION SU
PERIOR A MATRIZ";AT VAL "12",VAL
 "9"; FLASH VAL "1";"PULSE COMAN
DO": LET L=P*VAL "21": LET N=COD
E "@": GO TO CODE "<"
 110 GO TO CODE "7"
 120 IF A$="b" OR A$="B" THEN RU
N
 140 LET C=CODE ""
 150 LET C=VAL "1"*(A$="i" OR A$
="I")+VAL "2"*(A$="l" OR A$="L")
+VAL "3"*(A$="s" OR A$="S")+VAL
"4"*(A$="e" OR A$="E")+VAL "5"*(
A$="c" OR A$="C")+VAL "6"*(A$="x
" OR A$="X")+VAL "7"*(A$="Y" OR
A$="y")+VAL "8"*(A$="r" OR A$="R
")+VAL "9"*(A$="m" OR A$="M")+VA
L "11"*(A$="v" OR A$="V")+VAL "1
2"*(A$="p" OR A$="P")+VAL "13"*(
A$="o" OR A$="O")
 290 IF C=CODE "" THEN PRINT #CO
DE "";AT CODE "",VAL "7"; BRIGHT
 VAL "1"; FLASH VAL "1";" COMAND
O ERRONEO ": PAUSE CODE "d": RET
URN
 300 GO SUB c*VAL "500"
 310 RETURN
 510 CLS : LET H=VAL "1": FOR N=
VAL "1" TO L: PRINT #VAL "4";AT
VAL "22",CODE "";T$(N): RANDOMIZ
E USR VAL "3190"
 515 IF N=VAL "22"*H THEN LET H=
H+VAL "1": PRINT #CODE ""; BRIGH
T VAL "1"; FLASH VAL "1";AT CODE
 "",VAL "5";"IMPRIMIENDO DOCUMEN
TO": COPY
 520 NEXT N
 525 IF N<VAL "22"*H AND N>=L TH
EN FOR Z=L+1 TO VAL "22"*H: PRIN
T #VAL "4";AT VAL "22",CODE "";"
 ": RANDOMIZE USR VAL "3190": NE
XT Z: PRINT #CODE ""; BRIGHT VAL
 "1"; FLASH VAL "1";AT CODE "",V
AL "5";"IMPRIMIENDO DOCUMENTO":
COPY
 530 CLS : RETURN
1005 INPUT BRIGHT VAL "1";"NOMBR
E A CARGAR? ";N$
1007 CLS : PRINT AT VAL "7",VAL
"7"; BRIGHT VAL "1"; FLASH VAL "
1";"CARGANDO DOCUMENTO"; FLASH C
ODE "";AT VAL "10",VAL "1";"PULS
E ""PLAY"" EN EL CASSETTE Y,";AT
 VAL "12",VAL "2";"ESPERE UN MOM
ENTO, POR FAVOR"
1020 LOAD N$ DATA A(): LOAD N$ D
ATA T$(): LET L=A(VAL "1"): LET
P=A(VAL "2"): GO SUB VAL "3500":
 RETURN
1505 INPUT BRIGHT VAL "1";"CON Q
UE NOMBRE SE GRABA "; LINE N$: I
F N$="" THEN LET N$="?"
1510 LET A(VAL "1")=L: LET A(VAL
 "2")=P: IF LEN N$>VAL "10" THEN
 LET N$=N$( TO VAL "10")
1520 SAVE N$ DATA A(): POKE VAL
"23736",CODE "ASN ": SAVE N$ DAT
A T$(): RETURN
2000 IF L=VAL "21"*p THEN PRINT
#CODE "";AT CODE "",VAL "8"; BRI
GHT VAL "1";"No cabe mas texto":
 PAUSE CODE "2": RETURN
2001 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR PARA  ELEGI
R LA DE INSERTAR  ";K: IF K<=COD
E "" THEN GO TO VAL "2001"
2003 GO SUB VAL "5505": INPUT BR
IGHT VAL "1";"DESPUES DE QUE LIN
EA SE INSERTA ";D: IF D>L THEN G
O TO VAL "2003"
2010 LET D=D+VAL "1": FOR N=L TO
 D STEP VAL "-1": LET T$(N+VAL "
1")=T$(N): NEXT N: LET L=L+VAL "
1"
2040 INPUT BRIGHT VAL "1";"TEXTO
 A INSERTAR? "; LINE T$(D): CLS
: LET K=D-VAL "19": IF K<=CODE "
" THEN LET K=VAL "1"
2050 GO SUB VAL "4504": RETURN
2510 LET C=CODE "@"
2520 IF T$(L-VAL "1",C)=" " AND
C<>VAL "1" THEN LET C=C-VAL "1":
 GO TO VAL "2520"
2530 LET T$(L-VAL "1")=B$( TO (C
ODE "@"-C)/VAL "2")+T$(L-VAL "1"
): PRINT #VAL "4";AT VAL "21",CO
DE "";T$(L-VAL "1"): RETURN
3010 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR PARA  ELEGI
R LA DE BORRE";K: IF K=CODE "" T
HEN RETURN
3030 GO SUB VAL "5505": INPUT BR
IGHT VAL "1";"LINEA A BORRAR ";E
: IF E<=CODE "" OR E>L THEN CLS
: RETURN
3040 FOR N=E TO L: LET T$(N)=T$(
N+VAL "1"): NEXT N: LET L=L-VAL
"1": CLS : LET K=E-1: GO SUB VAL
 "4504": RETURN
3500 CLS : PRINT #VAL "4";AT COD
E "",CODE "";: FOR H=VAL "1" TO
L: FOR Y=VAL "1" TO CODE "@": PR
INT #VAL "4";T$(H,Y);: NEXT Y: N
EXT H: PRINT ;: PRINT #CODE "";A
T VAL "1",CODE ""; BRIGHT VAL "1
";" FIN DEL TEXTO, PULSA UNA TEC
LA ": PAUSE CODE "": RANDOMIZE U
SR VAL "3190": RETURN
4001 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR PARA  ELEGI
R LA DE REEMPLAZAR ";K: IF K<=CO
DE "" THEN GO TO VAL "4001"
```

```
4003 GO SUB VAL "5505": INPUT BR
IGHT VAL "1";"LINEA A REEMPLAZAR
? ";E: IF E>L OR E<=CODE "" THEN
 CLS : RETURN
4010 INPUT BRIGHT VAL "1";"TEXTO
 A REEMPLAZAR? "; LINE R$: INPUT
 BRIGHT VAL "1";"NUEVO TEXTO? ";
 LINE S$: FOR M=VAL "1" TO CODE
"@"-LEN R$: IF T$(E,M TO M+LEN R
$-VAL "1")=R$ THEN GO SUB VAL "4
080"
4050 NEXT M: CLS : LET K=E: GO S
UB VAL "4504": RETURN
4080 IF M=VAL "1" THEN LET T$(E)
=S$+T$(E)(LEN R$+VAL "1" TO )
4090 IF M<>VAL "1" THEN LET T$(E
)=T$(E)( TO M-VAL "1")+S$+T$(E)(
M+LEN R$ TO )
4100 RETURN
4501 CLS : LET K=VAL "1": GO TO
VAL "4504"
4503 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR? ";K
4504 LET H=VAL "1"
4505 FOR N=K TO L: PRINT #VAL "4
";AT VAL "21",CODE "";T$(N): RAN
DOMIZE USR VAL "3190": IF N=VAL
"21"*H+(K-VAL "1") THEN LET H=H+
VAL "1": PRINT #CODE "";AT CODE
"",CODE ""; BRIGHT VAL "1";"PULS
E ""0"" PARA ENTRAR MAS TEXTO U
OTRA TECLA PARA CONTINUAR       ":
 PAUSE CODE ""
4516 IF INKEY$="0" THEN CLS : GO
 TO CODE "7"
4520 PRINT #CODE "";AT CODE "",C
ODE "";B$+B$: NEXT N: RETURN
5501 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR? ";K: IF K<
=CODE "" THEN GO TO VAL "5501"
5520 CLS : LET H=VAL "1": FOR N=
K TO L-VAL "1": PRINT #VAL "4";A
T VAL "21",CODE "";N;":";T$(N);:
 RANDOMIZE USR VAL "3190"
5521 IF N=VAL "21"*H+(K-VAL "1")
 AND C=VAL "4" THEN LET H=H+VAL
"1": PRINT #CODE ""; BRIGHT VAL
"1";AT CODE "",CODE "";"PULSE:
""I"" PARA INSERTAR LINEA.";AT V
AL "1",VAL "8";"U OTRA TECLA PAR
A SEGUIR": PAUSE CODE ""
5522 IF N=VAL "21"*H+(K-VAL "1")
 AND C=VAL "6" THEN LET H=H+VAL
"1": PRINT #CODE ""; BRIGHT VAL
"1";AT CODE "",CODE "";"PULSE ""
X"" PARA BORRAR LA LINEA  U OTRA
 TECLA PARA SEGUIR.": PAUSE CODE
 ""
5523 IF N=VAL "21"*H+(K-VAL "1")
 AND C=VAL "8" THEN LET H=H+VAL
"1": PRINT #CODE ""; BRIGHT VAL
"1";AT CODE "",CODE "";"PULSE  "
"R"" PARA REEMPLAZAR TEXTO";AT V
AL "1",VAL "8";"U OTRA TECLA PAR
A SEGUIR": PAUSE CODE ""
5524 IF N=VAL "21"*H+(K-VAL "1")
 AND C=VAL "12" THEN LET H=H+VAL
 "1": PRINT #CODE ""; BRIGHT VAL
 "1";AT CODE "",CODE "";"PULSE
""C"" PARA CAMBIAR LINEA.";AT VA
L "1",VAL "7";"U OTRA TECLA PRA
SEGUIR": PAUSE CODE ""
5525 IF INKEY$="I" OR INKEY$="i"
 OR INKEY$="R" OR INKEY$="r" OR
INKEY$="C" OR INKEY$="c" OR INKE
Y$="X" OR INKEY$="x" THEN LET C=
VAL "11": RETURN
5526 IF N=VAL "21"*H+(K-VAL "1")
 THEN LET H=H+VAL "1": PRINT #CO
DE "";AT CODE "",CODE ""; BRIGHT
 VAL "1";"PULSE ""0"" PARA ENTRA
R MAS TEXTO U OTRA TECLA PARA CO
NTINUAR": PAUSE CODE ""
5527 IF INKEY$="0" THEN CLS : GO
 TO CODE "7"
5540 NEXT N: LET C=VAL "11": RET
URN
6001 INPUT BRIGHT VAL "1";"LINEA
 A PARTIR VISUALIZAR PARA  ELEGI
R LA DE CAMBIO  ";K: IF K<=CODE
"" THEN GO TO VAL "6001"
6005 GO SUB VAL "5510": INPUT BR
IGHT VAL "1";"LINEA A CAMBIAR? "
;Q: IF Q=CODE "" THEN CLS : RETU
RN
6020 INPUT BRIGHT VAL "1";"NUEVO
 TEXTO? "; LINE T$(Q): CLS : LET
 K=Q: GO SUB VAL "4504": RETURN
6510 CLS : GO SUB VAL "9270": CL
S : LET K=VAL "1": GO SUB VAL "4
504": RETURN
9270 CLS : PRINT AT CODE "",VAL
"7"; BRIGHT VAL "1";" C O M A N
D O S "''''';"M o m"; BRIGHT CODE
 "";" Visualiza el modelo"' BRIG
HT VAL "1";"X o x"; BRIGHT CODE
"";" Borra una linea"' BRIGHT VA
L "1";"Y o y"; BRIGHT CODE "";"
Leer texto en pantalla"' BRIGHT
VAL "1";"C o c"; BRIGHT CODE "";
" Centra la ultima linea"
9330 PRINT BRIGHT VAL "1";"E o e
"; BRIGHT CODE "";" Inserta una
linea"' BRIGHT VAL "1";"R o r";
BRIGHT CODE "";" Reemplaza palab
ra o frase"' BRIGHT VAL "1";"P o
 p"; BRIGHT CODE "";" Sustituye
una linea "' BRIGHT VAL "1";"V o
 v"; BRIGHT CODE "";" Muestra nu
meracion lineas"
9370 PRINT BRIGHT VAL "1";"I o i
"; BRIGHT CODE "";" Imprime el t
exto"' BRIGHT VAL "1";"L o l"; B
RIGHT CODE "";" Carga un texto"'
 BRIGHT VAL "1";"S o s"; BRIGHT
CODE "";" Graba un texto"' BRIGH
T VAL "1";"B o b"; BRIGHT CODE "
";" Borra el texto"' BRIGHT VAL
"1";"O u o"; BRIGHT CODE "";" Mu
estra las instrucciones"
9425 IF C=VAL "13" THEN PRINT #C
ODE "";AT VAL "1",VAL "1"; BRIGH
T VAL "1";"PULSE UNA TECLA PARA
RETORNAR": PAUSE CODE "": RETURN

9426 PRINT #CODE ""; BRIGHT VAL
"1";AT VAL "1",VAL "1";"CARGAR A
LGUN DOCUMENTO (S/N)? ": PAUSE C
ODE "": IF INKEY$="S" OR INKEY$=
"s" THEN GO SUB VAL "1005": LET
c=VAL "13"
9427 PRINT AT VAL "20",CODE "";
BRIGHT VAL "1";"PANTALLAS A RELL
ENAR ?-(max.25)-"
9430 INPUT P: IF P>VAL "25" OR P
<=CODE "" THEN GO TO VAL "9430"
9435 PRINT AT VAL "20",VAL "20";
 BRIGHT VAL "1";" ........ "; FL
ASH VAL "1";P
9440 PRINT #CODE "";AT VAL "1",V
AL "1"; BRIGHT VAL "1";"PULSE UN
A TECLA PARA COMENZAR."
9450 PAUSE CODE ""
9460 RETURN
```

```
        C O M A N D O S

M o m Visualiza el modelo
X o x Borra una linea
Y o y Leer texto en pantalla
C o c Centra la ultima linea
E o e Inserta una linea
R o r Reemplaza palabra o frase
P o p Sustituye una linea
V o v Muestra numeracion lineas
I o i Imprime el texto
L o l Carga un texto
S o s Graba un texto
B o b Borra el texto
O u o Muestra las instrucciones


PANTALLAS A RELLENAR ?-(max.25)-
```

TABLON

Cambio ZX SPECTRUM 48K más 200 programas (en C.M.) por ordenador MSX (RAM = 48K).
También poseo abundante bibliografía y programas en Basic.

Amador Merchán Ribera
C/ Cáceres, 8-3.° A
28045-Madrid. Telf. 467 48 14

Deseo intercambiar programas sin ningún fin económico. Tengo más de 100 en constante aumento. Enviar lista.

José María Castañeda Vercher
C/ Blasco Ibáñez, 24
Señera. 46200-Valencia

Vendo por 1.500 ptas. o cambio por algo interesante, curso de introducción a la informática y lenguaje Basic, ideal para estudiantes, profesionales y toda persona interesada en el apasionante tema, regalo una cinta con programas.

Apartado de Correos, 567
50080-Zaragoza

QUEREMOS VER TUS PROGRAMAS BASIC: SPECTRUM, C64, MSX Y AMSTRAD

Sabemos que eres lo suficientemente ingenioso para no necesitar copiarlos de nadie. Mándanos una cinta de cassette con tu programa, y a ser posible, un listado del mismo.

Premiamos con 5.000 ptas. y un ejemplar de la revista en cuestión, cada programa que publiquemos en cualquiera de nuestras cinco publicaciones.

Envíanos el tuyo, cuanto antes lo hagas, más posibilidades tendrás de verlo publicado.

MONSER, S.A.
C/ Argos, 9
28037-MADRID

Deprisa, deprisa...

Nos. 13 A y 13 B

sinclair - spectrum
48-K

SOFTWARE M AZINE

CON ESTE N O TRECE

Pídala en su kiosko o en tiendas especializadas

SUSCRIPCIONES: MONSER, S.A. C/ Argos, 9 - 28037 MADRID

2 S ETTES «FULL MEMORY 48K»
Y UNA CINTA VIRGEN



CON INSTRUCCIONES EN CASTELLANO

MENSUAL

MONSER





Con la marca Monser sobre el cassette, usted obtiene no solamente una excelente cinta para computador, sino también una cassette que le proporciona todas las funciones y conveniencias que requiera el usuario. El cassette para ordenadores personales Monser está diseñado para ser usado con microcomputadores y provee una combinación única de precisión y ejecución.

DE VENTA EN TIENDAS ESPECIALIZADAS.

Para envios dirigirse a Monser, S.A. c/ Argos nº 9. Tlf. 742 72 12 / 96.